加微信：642945106 发送“ 赠送 ”领取赠送精品课程
发数字“ 2 ”获取众筹列表

开篇词 | 把C++从“神坛”上拉下来，这次咱这么学

2020-05-06 罗剑锋

罗剑锋的C++实战笔记 进入课程 >

讲述：Chrono
时长 11:03 大小 10.13M

你好，我是罗剑锋，你叫我 Chrono 就好。

去年，我在极客时间开了一个《透视 HTTP 协议》的课程，有很多同学留言，希望能再听我讲讲其他领域的知识。

于是，在一年之后的今天，我给你带来了这个新课程：《罗剑锋的 C++ 实战笔记》。

为什么 C++ 这么难学？

如果你之前看过那个课程，就应该知道，我的工作经历比较杂，HTTP 只能算是我的一个“副业”。这次要讲 C++，感觉终于回到了“老本行”。毕竟写了二十多年的 C++ 代

码，经手的大大小小的 C++ 项目不计其数，现在终于有机会把一点一滴积累起来的这些经验整理、分享出来，内心还是有点激动的。

一说到 C++，几乎所有人的第一反应就是“出了名的难学难用”。的确如此，因为它实在是太复杂了，有太多的特性和细节。

随着标准版本的演进，C++ 里包含的东西也越来越多。最早的 C++98 只有 60 来个关键字，到 C++11 变成了 70 多个，C++20 则膨胀到了近百个。对比一下同级别的 Java、Go 等语言，C++ 真称得上是“巨无霸”。而且这还仅仅是核心语言，外面还有更庞大的标准库在等着你。

不断膨胀的核心语言加上庞大的标准库，让学习、使用 C++ 的门槛无形中提高了很多，不仅 C++“新手”学起来很难，就连 C++“老手”也会觉得，用好它并不是一件容易的事情。

Effective C++ 里有一句话，我觉得很有意思：

> C++ 是一个威力十足的编程语言，如果 C 带给你足够绞死自己的绳索，C++ 就是间五金店，挤满了许多准备为你绑绳结的人。

这句话形象地说出了 C++ 的难点：**它太接近底层，C 语言本身已经有很多“坑”了，而 C++ 又增加了更多的“坑”，一旦用不好，就很容易“作茧自缚”**。

其实，这些年来，C++ 标准委员会也意识到了这个难学难用的问题，也做了很多工作，尽量让 C++ 对初学者友好，朝着易学易用的方向去努力。但 C++ 毕竟背着“兼容 C 语言”这个巨大的历史包袱（说得重一点就是“原罪”），无法做出彻底的改革，在可以预见的将来，语言里的那些“坑”还将长期存在。

针对这个问题，我的建议是，**先从 C++11 标准开始学起**。这个版本的 C++ 虽然还是很复杂，但却添加了很多方便易用的新特性，更接近“现代编程语言”，可以少遇到一些传统编程方式的“坑”。

市面上有不少教授现代 C++ 的书，也都是专家、大师之作，权威性毋庸置疑。但 C++ 实在是太庞大了，相应的书都很厚，慢慢去“啃”、去“消化”实在是吃力。

而且，这些毕竟是纸面上的知识，离实际的开发还有一定的距离，你难免会有这样的感慨：

“**道理我都懂，可用起来还是会犯怵，要是身边能有个人来指点一下该多好。**”

不知道你在刚毕业的时候，公司有没有为你安排过一个“入职导师”的角色，他会制定培养计划，带你熟悉环境，指导你的工作，让你尽快成长为一名合格的职场新人。

C++ 书籍就好像是学校里的老师，只能教你基本的知识。而学习 C++ 最缺乏的就是一个“入职导师”，他能帮你跨越从课堂到现实的“鸿沟”，告诉你实际工作时会遇到哪些问题，又该怎么解决。

很可惜，大多数人，也包括我，当初都没有遇到这样的好导师，学 C++ 的时候一切都要靠自己摸索。虽然说“实践出真知”，最终有所成就，但也浪费了不少大好年华。

所以，接到极客时间的邀请之后，我决定写这样一个能够担当“入职导师”“引路人”角色的课程，从庞大的 C++ 里裁剪出一个精致的子集，挑选出最适合你自己的 C++ 特性。我还会把踩过的坑、走过的弯路、收获的果实，都毫无保留地分享给你。

防止断更 请务必加
首发微信：17161143665

课程特点

既然要当“入职导师”，那我的目标就是一切从实际出发，只讲实实在在、脚踏实地的 C++ 知识，而不会讲那些“高深”的理论和“玄乎”的技巧，更不会去教你那些“屠龙之术”。

另外，因为 C++ 的资料已经有很多了，我也不想变成标准规范的“复读机”，机械地重复那些接口定义。所以，在这个课程里，我通常只会简单提一下功能要点，不会详细解释调用方式，**重点是谈使用时的注意事项和经验教训**，具体怎么用你完全可以去查资料。

讲 C++ 必然要写代码，不过课程示例里的代码都很短，也不复杂，对 C++ 水平的要求很低，不需要你有太多的经验（1～5 年都可以），保证让你一眼就能看明白。虽然代码可以说是“玩具”，但里面蕴含的知识却绝不是“玩具”，这就需要你看懂之后去细心领会了。

总之，我想尽量降低这门课的学习门槛，把 C++ 从“神坛”上拉下来，让它平易近人一些，希望能够让你看到 C++ 也有亲切的一面。

在这里请允许我适当引用并修改《设计模式》一书里的部分文字，来描述一下这门课的特点：

> ……并不要求使用独特的语言特性，也不采用那些足以使你的朋友或者老板大吃一惊的神奇的编程技巧。

> ……有经验的 C++ 程序员的确能够做出良好的设计，写出优秀的代码，而新手则面对众多选择无从下手，需要花费较长时间领会良好的 C++ 代码是怎么回事。有经验的 C++ 程序员显然知道一些新手所不知道的东西，这又是什么呢？

> ……课程里不会提出任何前所未见的新算法或者新程序设计技术，既没有给出一种严格的系统设计方法，也没有提出一套新的设计理论——它只是将现有的一些经验加以文档化。

> ……一旦你理解了 C++，并且有了一种“Aha！”（而不是“Huh？”）的应用经验和体验后，你将用一种非同寻常的方式思考 C++ 编程。

课程设计

按照这个思路，我把我最有切身感受、最有实际意义的经验，全部浓缩在了这个课程里。学会了这些“武艺”，你一定能够用 C++ 开发出优雅、高效的程序。

整个课程分为五个模块，**注重语言和库的“开发落地”，基本不讲语法细节和内部实现原理，而是用实例促使你更多地应用“现代 C++”自然、直观的思维方式**。

C++ 与 C 是一脉相通的，很多时候，C++ 不过是 C 的高级解法。所以，即使你的主力工作语言是 C，也可以过来看看，了解一下新思路、新工具。

我先给你大概介绍一下这些模块吧。

在“**概论**”模块，我会从程序的**生命周期**和**编程范式**这两个独特的角度来审视它，帮你看清楚 C++ 复杂的本质，透彻理解 C++ 程序的运行机制和面向对象编程思想。

在“**语言特性**”模块，我精选出了 C++ 中的**自动类型推导**、**智能指针**、**Lambda 表达式**等几个重要特性，帮你掌握惯用法，消灭代码里的隐患，用这些特性写出清晰、易读、安全

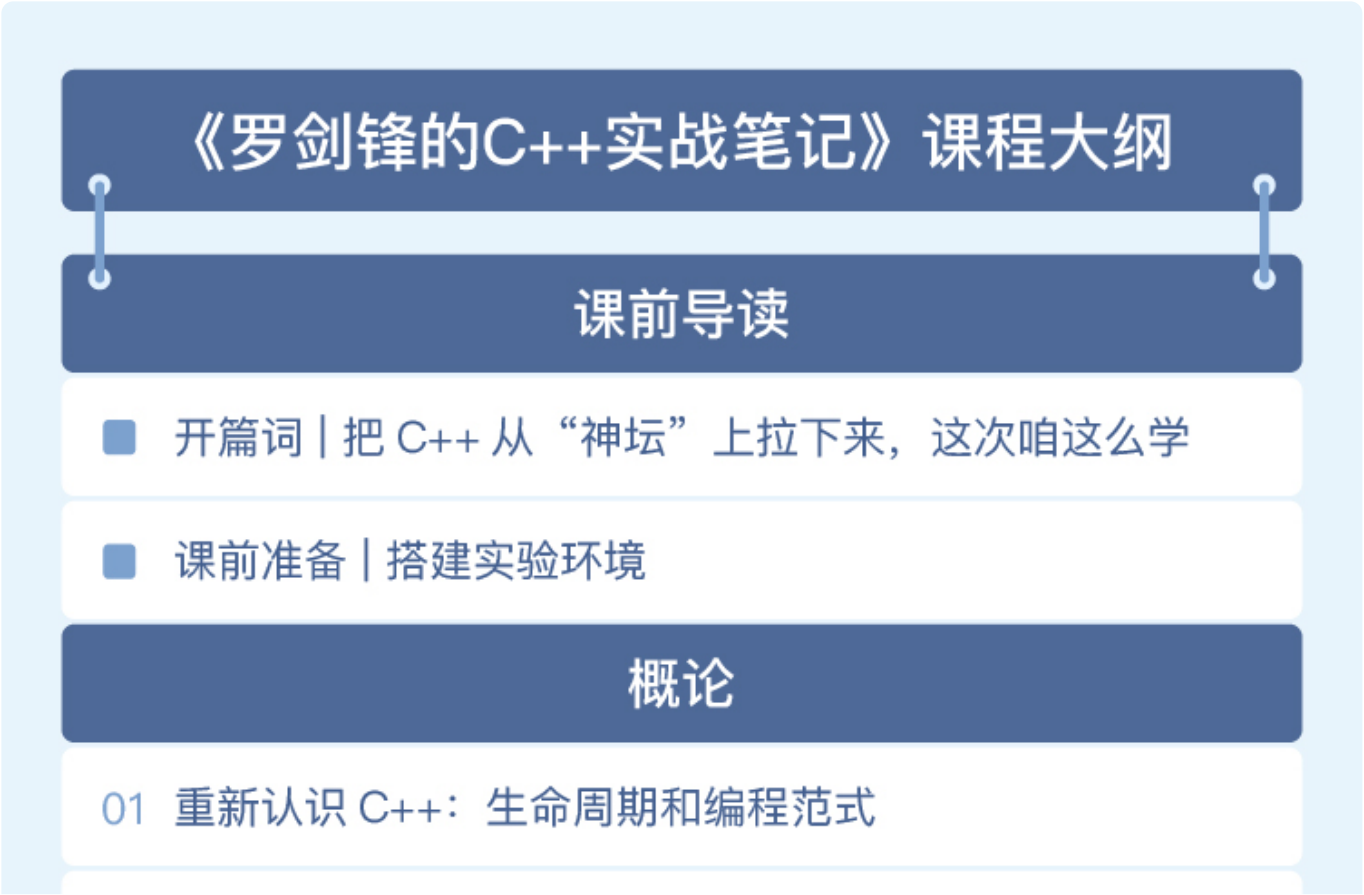
的代码。

标准库是 C++ 里占比非常大的一部分，重要性不亚于语言本身。所以在“**标准库**”模块，我会介绍其中最核心的四个部分：**字符串、容器、算法和并发**，让你用好这个最基本的库，学会泛型编程，提高程序的运行效率。

不过，标准库也不可能涵盖所有的开发领域，所以在“**技能进阶**”模块里，我会介绍 C++ 标准之外的一些第三方工具，带你一起去实现序列化、网络通信和性能分析等功能，解决实际开发中遇到的常见问题。

之后是“**总结**”模块，我会结合 C++ 讲讲设计模式，并给出一个完整可用的 C++ 服务端程序例子（这里会与《透视 HTTP 协议》这门课有个小小的联动）。这样“理论结合实际”，把前面的所有知识点都串联起来，让你看看在项目中 C++ 是具体怎么思考、设计、落地的。你实际动手研究一下代码，再试着改改，就能够把 C++ 的这些特性融会贯通了。

除此之外，我还特别设计了一个“**轻松话题**”单元，和你聊些 C++ 之外的东西，以避免因为课程安排得太紧凑，没有“喘息”的机会，让你学起来很累。这些话题涵盖的范围比较广，包括经典的学习资料、提高工作效率的工具等，让你在掌握核心硬技能的同时向外拓展知识面，“会工作，更要会生活”。



02 编码阶段能做什么：秀出好的 code style
03 预处理阶段能做什么：宏定义和条件编译
04 编译阶段能做什么：属性和静态断言
05 面向对象编程：怎样才能写出一个“好”的类?

语言特性

06 auto/decltype：为什么要有自动类型推导?
07 const/volatile/mutable：常量 / 变量究竟是怎么回事?
08 smart_ptr：智能指针到底“智能”在哪里?
09 exception：怎样才能用好异常?
10 Lambda：函数式编程带来了什么?

标准库

11 一枝独秀的字符串：C++ 也能处理文本?
12 三分天下的容器：恰当选择，事半功倍
13 五花八门的算法：不要再手写 for 循环了
14 十面埋伏的并发：多线程真的很难吗?

技能进阶

15 序列化：简单通用的数据交换格式有哪些?

16 网络通信：我不想写原生 Socket
17 脚本语言：搭建高性能的混合系统
18 性能优化：该用什么工具？怎么用？
总结篇
19 设计模式（上）：该怎么学设计模式？
20 设计模式（下）：C++ 里的设计模式
21 知识串讲（上）：带你开发一个书店应用
22 知识串讲（下）：带你开发一个书店应用
期末测试 | 这些 C++ 核心知识，你都掌握了吗？
轻松话题
轻松话题（一）| 经典的技术书单大放送
轻松话题（二）| 给你分享我的工作百宝箱
轻松话题（三）| 学习之余，来读点儿不一样的
轻松话题（四）| 提高生活质量的手机 App
结束语 | The Road Ahead

学前勉言

在开课之前，我还想和你分享几句编程格言。这三条格言已经陪伴了我很久，一直指导着我的编程实践。

> 任何人都能写出机器能看懂的代码，但只有优秀的程序员才能写出人能看懂的代码。

> 有两种写程序的方式：一种是把代码写得非常复杂，以至于“看不出明显的错误”；另一种是把代码写得非常简单，以至于“明显看不出错误”。

> “把正确的代码改快速”，要比“把快速的代码改正确”，容易得太多。

C++ 庞大、复杂是无法改变的事实，所以我们要把这三条格言铭记在心，对它保持一颗“敬畏”的心，在学习语言特性的同时，千万不要滥用特性，谦虚谨慎，戒骄戒躁。

我很喜欢 15 年前乔布斯在斯坦福大学演讲中的一句话，觉得非常适合 C++。所以，最后我想把它送给你，我们共勉，希望在接下来的这段时间里，我们一起：

Stay Hungry, Stay Foolish.

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

精选留言 (34)

写留言

小林coding
2020-05-06

知道的越多，不知道的也越多。一年精通C++，3年熟悉C++，6年了解C++，10年用过C++

作者回复: 说得非常好，所以学C++的人都虚怀若谷，心胸宽广。

3 14

吃草🐂~
2020-05-06

大学里学过 C++，然而只是皮毛，所以一直只知道难，但是不知道难在哪里。目前已是三年半开发经验的 Java 程序猿~个人觉得 Cpp 更贴近底层，并且也想了解这门语言的一些设计思想。于是乎，果断入手开始研究~🤓🤓

展开

作者回复: 就是要有这样的钻研态度，学C++，不一定非要用它做开发，而是把它当做他山之石，开拓自己的知识面。

空闲时间可以自己写一些小程序，用来练习C++，熟悉系统底层。

5

Carlos
2020-05-06

哈哈，我刚在朋友圈宣布自己学会了c++, 这套课程就来了，完美衔接！

作者回复: 欢迎欢迎。

3

蓝配鸡
2020-05-06

看到是罗老师， 先买了一定不亏😂

展开

作者回复: 多谢支持，老朋友了。

3

秦奋
2020-05-06

老师真是很亲切，我看留言都回复了呢。加油

展开

作者回复: 是啊，有很多同学都说我是回复得最勤快的人。

2

rainnnnn
2020-05-09

买这个课的理由：1、我想学好C++，2、这个课的老师是罗剑锋。

作者回复:
1.欢迎加入大家庭一起学习。

2.受宠若惊，实在是不敢当。

1

徐有志
2020-05-06

Stay Hungary, Stay Foolish.

展开

作者回复: 这句话有很多种理解，所以才含义深刻。

3 1

yelin
2020-05-11

加油

展开

作者回复: welcome to C++ world.

Lincoln
2020-05-10

马上就要去鹅厂实习了c++后台 结果我c++还不会 慌了慌了

作者回复: 不用慌，在实践中学习，功力涨得更快。

姜海天
2020-05-09

保持一颗敬畏之心，摆正态度，“简单”地调用C++，适当地向C++妥协，能够让以后的自己看懂代码，让别人看懂代码，这样，能够最大限度发挥计算机的作用，更好造福人类。

不应该为了耍酷而使用炫技，看似酷炫，实则害人害己，“结硬寨，打呆仗”，稳中求...
展开

作者回复: 说的很好，写代码的最终目的是给人看，而不是给机器看。

或许 没有
2020-05-09

研究生才开始学习C++，发现C++在很多地方都在更新，但是发现多得让人头晕了，之前版本成立的可能在新版本中被否定。头大
展开

作者回复: 是的，所以我建议现在从C++11/14开始学起，太早的C++98和太新的C++17/20都不要看，不要一味追新。

11/14已经很“现代”了，可以写出媲美Java、Python的优雅代码，学习难度也低了很多，编译器的支持也是最好的。

longbin
2020-05-09

正在学习，有录音真实太亲民了，路上也能听录音学习提高编程能力啦

作者回复: 欢迎一起学习C++。

默
2020-05-08

我的C+编程能力还是小白，通过您的课程学习，之后我还需要做些什么才能更快的提高呢。

作者回复: 多看一些开源项目，比如Nginx、leveldb、folly，然后关键是要在实践中用。

BulkBulky
2020-05-08

我工作中主要语言是python， 想了解下C++。想请老师介绍下懂得C++对平时使用python有何帮助?

作者回复: 后面会专门讲混合编程，用C++写底层模块，让Python调用，提高运行效率。

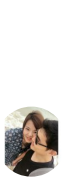
C家族铁粉
2020-05-07

看到罗老师说课程中会有一个完整可用的服务端的例子，能否稍微具体点介绍下这个例子呢？感觉设计模式和并发编程只用一两篇文章似乎不太够啊，要是罗老师以后能结合自己的感悟，出专栏从零开始讲讲这两个大主题就更好了，并且采用C/C++语言来教学。

展开

作者回复: 代码都放在了GitHub上，可以直接看。

你说的很对，这些都是很大的话题，但这个专栏的主题是C++，所以不可能把这些讲深讲透，只能结合C++讲几个要点，希望以后能有机会深挖吧。



baoqger
2020-05-07

我是web背景，做了三年了，想系统学c++，再想办法跟web背景整合下，比如nodejs，web assembly

作者回复: 想法很好，node.js好像就是用C++开发的，也可以选择Nginx/C/C++/Lua。

微风漂过
2020-05-07

STL和C++多线程应该如何学习？
另外，跨平台的C++开发应该如何学习？

作者回复:
1.这两部分会在后面的标准库里面讲，也可以先找经典C++书籍提前了解一下。

2.C++语言是跨平台的，但涉及到底层系统编程，就需要了解底层的细节了，可以用宏、条件编译、包装外观等模式来屏蔽差异。
这方面Nginx、Boost都是很好的示例，可以参考。

涤生
2020-05-07

学好一门系统底层语言应该不亏

展开

作者回复: 没错，即使不靠C++吃饭，对于计算机、互联网从业人员都是非常有用的。

pcdiy

2020-05-06

以前学过C，但是一头雾水，不明白是什么，感觉很难懂，但还是选择学，不管懂与不懂

作者回复: 如果是零基础学C、C++，可能还是要了解一下计算机组成原理，毕竟它们接近系统底层，与硬件打交道比较多。

不过现在的C++11已经改善了很多，如果你会其他的编程语言，一定能够在C++里找到自己熟悉的那部分。

石头

2020-05-06

算是零基础学习C++，之前一直是C开发。

展开

作者回复: 一开始可以把C++当成是C with Class，了解类和封装，然后再慢慢学习高级特性，模板、泛型这些还是有点复杂，刚开始可以不看，找出C++里最合适自己的那部分来学。